# HLT Configuration(s) and “other business”

Updated version of
http://indico.cern.ch/getFile.py/access?contribId=95&sessionId=9&resId=0&materialId=slides&confId=44134

# HltLine (concept)

- An Hlt decision involves more than a single algorithm
  - one might pick some L0 candidates
  - another (sequence) may do some VELO reconstruction
  - a third may match the above two
  - ....
- In addition, it has some 'standard' pre/post processing:
  - 'entry' filters (ODIN, and/or L0, and/or other HLT decision(*))
  - Prescale
  - "The algorithm (sequence) which most people think of as the decision"
  - Postscale
- And it needs to record the final yes/no result
- And possibly catch errors in the 'hosted' algorithms
- Let's call this entire structure an 'HltLine'

(*) why the small print: because it breaks an important rule: HltLines should be *independent* of each other
Unfortunately, we need some exceptions: Hlt1Global (the OR of all Hlt1Lines) by definition depends on other lines
exception on the exception: Would be OK for an Hlt2 line to depend on an Hlt1 line

# HltLine (C++)

- Invokes the various stages
  - Each stage is an independent algorithm, HltLine *only* relies on 'FilterPassed'
- Updates an entry in HltDecReports 'as it goes along'
  - Catches exceptions and errors in the 'hosted' algorithms, updates HltDecReport entry accordingly, and recovers (hopefully)
  - As a result, the status is recorded in TES (in HltDecReport**s**) in uniform and reliable way:
    - convention: *all* configured Hlt1 decisions appear in HltDecReport**s**, regardless of their result
    - at some point we may remove negative decision before conversion to rawbank -- but only after it has been shown that on readback we can reliably recreate the (relevant) missing information from the configuration
  - Accept/Reject is recorded separately from 'how far we got': even if the event 'fails', it could be accepted, eg. because HltLine caught an exception: a (limited!) number of such errors will/could result in an 'accept'.

# HltLine (python)

Hlt1Lines are created by calling some dedicated python code

- Enforces uniform naming convention
  - might add some additional rules!
- Makes it easier to write Hlt1Lines
  - for HltAlgorithms, automatically 'daisy chains' input and output
    - TODO: improve management/specification of 'many-to-one' flow
  - Can 'copy-and-modify' entire lines to easily changes create variations:
    - eg. clone, decrease threshold and increase prescale
  - re-use pre-defined sequences (of sequences, of ... ) through 'bindMembers'
    - usefull, as each line *MUST BE* independent, and can not rely on results of other lines
- Registers the existence of a line -- this is used to actually configure Hlt1

*If you want to add an Hlt1 decision, you MUST write an Hlt1Line*

# Example

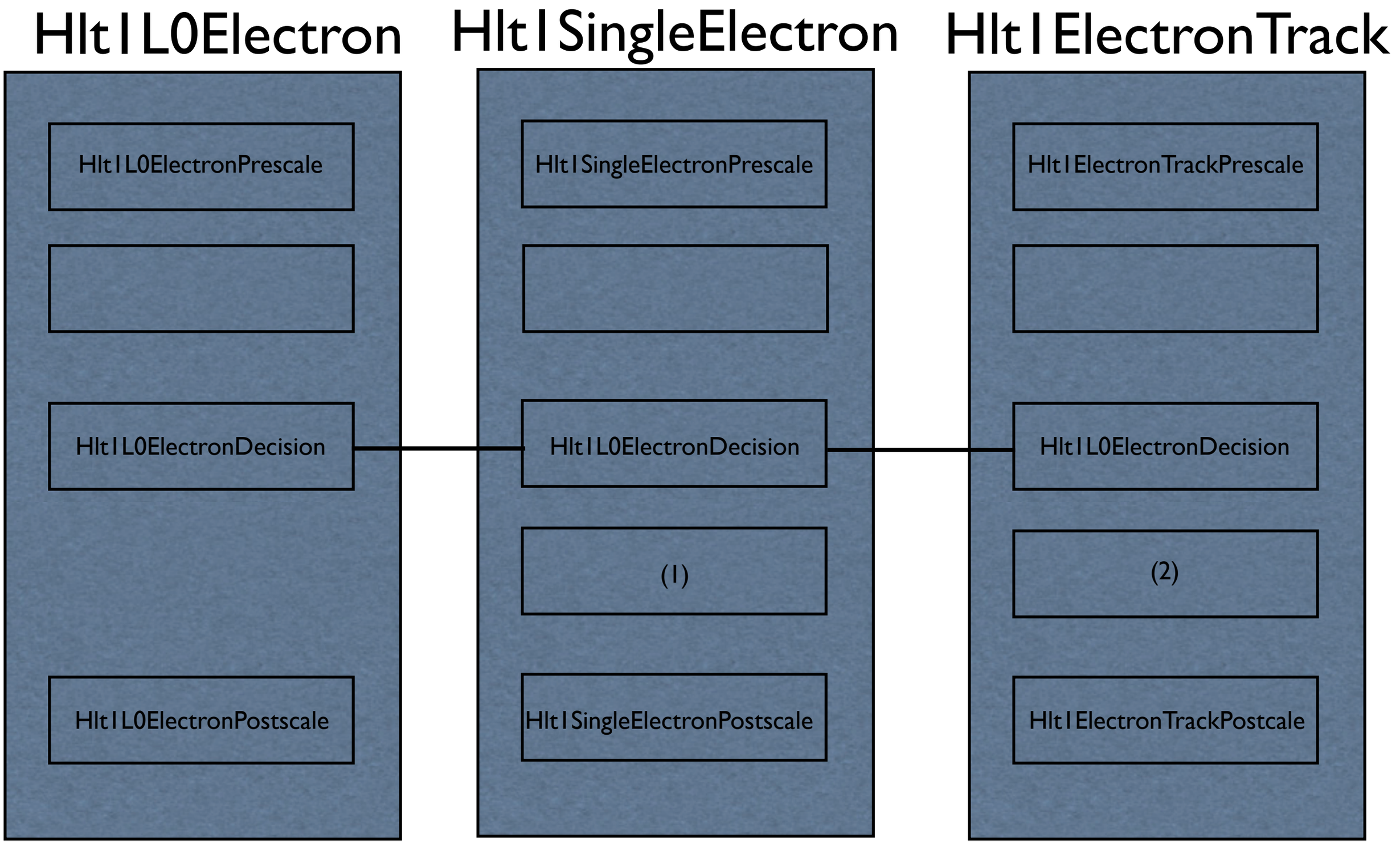
```
class HltL0LinesConf(LHCbConfigurableUser) :
   __slots__ = { 'Prescale'  : 1
                 , 'Postscale' : 0.000001
                 , 'L0Channels' : []  # if empty, use all pre-defined channels
                 }

   def __apply_configuration__(self):
      channels = self.getProp('L0Channels')
      if not channels : channels = L0Channels()
      for channel in channels :
          Line ( 'L0' + channel
               , L0DU  = "L0_CHANNEL('"+channel+"')"
               , prescale = self.getProp('Prescale')
               , algos = [ convertL0Candidates(channel) ]
               , postscale = self.getProp('Postscale')
               )
```

If an L0 TCK is explicitly specified, HltConf().L0TCK = ... goes out and finds the configured L0 channels, otherwise default to the 'canonical' list of L0Channels

Note: HltConf().L0TCK does *NOT* configure the L0DU to use this TCK, it is intended to be set by the configuration layer 'above' HltConf which *also* actually configures L0...
It is there such that at configuration time, one can check that only L0 channels which exist get used, and allow to check (partly) the consistency between L0 & HLT

Insures that all users of the L0 candidates of a given L0 Channel use the same algorithm instance to convert them. This way, this only gets done once, while keeping the lines independent




Hlt1L0ElectronDecision runs *at most* once per event, and (1) and (2), when executed, can rely on it always having run before (1) or (2) is excuted, i.e. all three lines are independent by construction!

# More complicated...

```
IP_CUT  = str(self.getProp('Ele_IPCut'))
PT_CUT = str(self.getProp('Compan_PtCut')
companionTrackWithIP  = [ RecoVelo
                        , Member ( 'TF', 'CompanionVelo',
                                   FilterDescriptor = [ 'IP_PV2D,|||[],'+IP_CUT+',3.', 'DOCA_%TFVeloT,<,0.2' ])
                        , Member ( 'TU', 'CompanionForward', RecoName = 'Forward' )
                        , Member ( 'TF', 'CompanionForward',  FilterDescriptor = ['PT,>,'+PT_CUT])
                        , Member ( 'VM2', 'DiElectron',
                                   InputSelection1  = '%TFVeloT',  InputSelection2 = '%TFCompanionForward',
                                   FilterDescriptor = ['DOCA,<,0.2'])
                        ]
Line( 'ElectronTrackWithIP'
        , L0DU = "L0_CHANNEL('Electron')"
        , algos = [ convertL0Candidates('Electron') ] + prepareElectronWithIP + companionTrackWithIP
              + [ Member ( 'VF', 'VertexCut'
                         , FilterDescriptor = [ 'VertexPointing_PV2D,<,0.5', 'VertexDz_PV2D,>,0.' ]
                         , OutputSelection = '%Decision') ]
      )

Line( 'ElectronFromPi0TrackWithIP'
        , L0DU = "L0_CHANNEL('LocalPi0')"
        , algos = [ convertL0Candidates('LocalPi0') ] + prepareElectronWithIP + companionTrackWithIP
              + [ Member ( 'VF', 'VertexCut'
                         , FilterDescriptor = [ 'VertexPointing_PV2D,<,0.5', 'VertexDz_PV2D,>,0.' ]
                         , OutputSelection = '%Decision') ]
      )
```

# HltLine (python)

- Hlt1Lines are grouped together ('alleys')
  - PA(ss through), L0, VE(lo), LU(minosity), MU(ons), HA(drons), EL(ectrons), PH(otons), B(eam)G(ass)
    - mu+track is part of MU
    - One .py file *per alley* defines *all* lines in that 'alley'
- Configuration of Hlt1Lines is triggered through HltConf().hltType = ...
  - Hlt1 == L0+VE+LU+MU+HA+EL+PH
  - if Hlt2 in this + seperated list, the Hlt2 configuration is invoked after Hlt1 -- so you can specify Hlt1+Hlt2 (the default)
  - HltType 'configures' lines, *then* they get 'added' to the Hlt1 sequencer
    - this is in a postconfig step (which has confused some)
    - because there is some (small) order dependence (eg. Hlt1Global and Hlt1IgnoringLumi can only be made after the rest)

# TCK — Trigger Configuration Key

- The Python code is used to *generate* configurations (by running a careful configured Moore), which are then stored in a 'database'
  - 'master' is in CVS (Hlt/TCKData), can copy to any CORAL supported database (SQLite, Oracle, ... )
  - new configurations become available for use through new releases of the Moore project (TODO: move Hlt/TCKData to DBase so that eg. DaVinci can access this information)
- The individual configurations are identified by 128 bit configuration IDs (think of this ID as the checksum of the HLT part of the configuration)
- TCK 'is a' 32 bit value which labels configurations, lower 16 bits specify L0 config.
- The 'online' Hlt (Moore) is configured from the above database
  - initial TCK -- the one to be used during initalization -- is specified by run-control to avoid waisting time on first event
  - Moore 'preloads' all other configurations of the same type as the initial one, and allows configuration updates within this set during fast run change
  - before processing each event, a svc insures that Hlt configuration matches TCK in ODIN bank
- Utilities exist for querying this database

# TCKsh

- SetupProject Moore
- TCKsh
  - is just: python -i -c 'from TCKData.utils import *'
- can list configurations, compare them, query for properties of algorithms, etc.

- listConfigurations()
  - shows all known configurations, grouped by release, and 'hltType'

- listAlgorithms(0x803b0000)
  - given either a TCK or a configID, shows a list of configured algorithms

# getHlt1Lines

- returns a list of Hlt1 lines

```
>>> from pprint import pprint
>>> pprint( getHlt1Lines(0x803b0000) )
['HltLine/Hlt1Physics',
 'HltLine/Hlt1Random',
 'HltLine/Hlt1Incident',

 'HltLine/Hlt1L0Muon',
 'HltLine/Hlt1L0DiMuon',
 'HltLine/Hlt1L0MuonNoGlob',
 'HltLine/Hlt1L0Electron',
 'HltLine/Hlt1L0Photon',
 'HltLine/Hlt1L0Hadron',
 'HltLine/Hlt1L0LocalPi0',
 'HltLine/Hlt1L0GlobalPi0',

 'HltLine/Hlt1VeloASide',
 'HltLine/Hlt1VeloCSide',
 'HltLine/Hlt1VeloClosing',

 'HltLine/Hlt1LumiNoBeam',
 'HltLine/Hlt1LumiBeamCrossing',
 'HltLine/Hlt1LumiSingleBeamRight',
 'HltLine/Hlt1LumiSingleBeamLeft'

 'HltLine/Hlt1XPress',

 'HltLine/Hlt1SingleHadron',
 'HltLine/Hlt1DiHadron',

 'HltLine/Hlt1MuonNoIPL0',
 'HltLine/Hlt1MuonNoIPGEC',
 'HltLine/Hlt1MuonIPCL0',
 'HltLine/Hlt1MuonIPCGEC',
 'HltLine/Hlt1DiMuonNoIPL0Di',
 'HltLine/Hlt1DiMuonIPCL0Di',
 'HltLine/Hlt1DiMuonNoIP2L0',
 'HltLine/Hlt1DiMuonNoIPL0GEC',
 'HltLine/Hlt1DiMuonIPC2L0',
 'HltLine/Hlt1DiMuonIPCL0GEC',
 'HltLine/Hlt1DiMuonNoIPL0Seg',
 'HltLine/Hlt1DiMuonNoIPGECSeg',
 'HltLine/Hlt1DiMuonIPCL0Seg',
 'HltLine/Hlt1DiMuonIPCGECSeg',
 'HltLine/Hlt1MuTrack',
 'HltLine/Hlt1MuTrack4JPsi',

 'HltLine/Hlt1SingleElectron',
 'HltLine/Hlt1SingleElectronFromPi0',
 'HltLine/Hlt1ElectronTrackWithIP',
 'HltLine/Hlt1ElectronFromPi0TrackWithIP',
 'HltLine/Hlt1ElectronTrackNoIP',
 'HltLine/Hlt1ElectronFromPi0TrackNoIP',

 'HltLine/Hlt1Photon',
 'HltLine/Hlt1PhotonFromEle',

 'HltLine/Hlt1IgnoringLumi',
 'HltLine/Hlt1Lumi',
 'HltLine/Hlt1Global']
>>>
```

PA

L0

VE

LU

XP

HA

MU

EL

PH

Autogenerated

```
>>> listAlgorithms(0x803b0000)
   Hlt                                                    01f182dae96749445aa469a8f032dd70
      Hlt1                                                9b9b5a478af8884e80fbfc1864a99714
         Hlt1Physics                                      d6bb6d69e435380360255f6912351377
            Hlt1PhysicsPreScaler                          ca9ed1f32c09145df08bbf71d36ba50d
            Hlt1PhysicsODINFilter                         b9e363e8b977c8feb0cbcbabd651c483
            Hlt1PhysicsPostScaler                         91c2849f094b13f25c332475f370618b
         Hlt1Random                                       4c2a781abc0699a25246bc3e27804418
            Hlt1RandomPreScaler                           79784825e61c710366fa6227b410a371
            Hlt1RandomODINFilter                          fd3e192285139ed70a88b6ffbc2fa7ee
            Hlt1RandomPostScaler                          793d9ca6443b12b99317f22fda868b7a
         Hlt1Incident                                     2017f251836ffa8f55d70a997d04f789
            Hlt1IncidentPreScaler                         89b0468c2a3b9810ee18d1f2bc663ee4
            Hlt1IncidentODINFilter                        60a457436ea07e2e7a38e25916330b3a
            Hlt1IncidentFilterSequence                    8b2e592469570e0fb7a48e7165abb583
               HltIncidentFilter                          1e4e6bbdaa64a2d5eac2afb40f46f5c9
            Hlt1IncidentPostScaler                        1c5226b5bd3a5466db59ed5ca1815d1f
         Hlt1L0Muon                                       f772142542ed8c052dc632ce6e96e7d7
            Hlt1L0MuonPreScaler                           9f201dba03c1f65a9a12a340b8fda898
            Hlt1L0MuonL0DUFilter                          7cf5cfcf1e950078ea3e6469de7d06ad
            Hlt1L0MuonFilterSequence                      09f36ef0fcd1ec0d299ba75ce1f47ec3
               Hlt1L0MuonDecision                         7fa7fef784853ea74c66ba035d72c89c
            Hlt1L0MuonPostScaler                          21248afe01abc96c58de824084990407
         Hlt1L0DiMuon                                     6e5289f83ca04106084561d73ff3e922
            Hlt1L0DiMuonPreScaler                         eb2c3d554d9cfb3bd9a32c5ba29001b8
            Hlt1L0DiMuonL0DUFilter                        75ab38a76eeeec0a2854ba1f448c33b4f
            Hlt1L0DiMuonFilterSequence                    c164e8eb96dd3f721dbc54b9c3624350
               Hlt1L0DiMuonDecision                       417e6461546e2cb7f4456325e32ff99f
            Hlt1L0DiMuonPostScaler                        fad58bf232d295bbf085351211a5f642
         Hlt1L0MuonNoGlob                                 e10051fec060efc35829866d8a5aba0e
            Hlt1L0MuonNoGlobPreScaler                     75d9d5327a07d5133e0bf422c9236225
            Hlt1L0MuonNoGlobL0DUFilter                    1985ba0cfe55f7ae22989107ca6a95ae
            Hlt1L0MuonNoGlobFilterSequence                3fa87d6e9324fa232d58a058e250eadf
               Hlt1L0MuonNoGlobDecision                   5ba26ec3383030abf33b2794a4214fa0
            Hlt1L0MuonNoGlobPostScaler                    a8e6249dc2fded058eb5c36f01dc805f
         Hlt1L0Electron                                   1610ca891bfadeee40143f14364b1d7f
            Hlt1L0ElectronPreScaler                       7906c5271f6c5877ba5cc8fed00dcfff
            Hlt1L0ElectronL0DUFilter                      13dd720458898e1afb61879b597c1ae7
            Hlt1L0ElectronFilterSequence                  7da598c63730331b54956278e02b79bc
               Hlt1L0ElectronDecision                     b3215d771cabb8846010184bf6b81b1d
            Hlt1L0ElectronPostScaler                      69670ba1d3b2ae9d7d34ad8f08d4455c
         Hlt1L0Photon                                     a40ae6f37c75b0d727f8363915b3a2b8
            Hlt1L0PhotonPreScaler                         eda84958cfd9d19fe5a01f339d17311c
            Hlt1L0PhotonL0DUFilter                        c73d81cab93dabac65f10a3a0c2e84d3
            Hlt1L0PhotonFilterSequence                    b2f72c8162094f69caf000a5b23dd3d0
               Hlt1L0PhotonDecision                       aa05db2e92fd3d24d73465782141882b
            Hlt1L0PhotonPostScaler                        1ce5a625598c6d95914c470d00bdd1a3
         Hlt1L0Hadron                                     d99ccc4f9e9c5553ca0cb14c9ec94712
            Hlt1L0HadronPreScaler                         685939bfcc38171c516f7e6d80aa592f
            Hlt1L0HadronL0DUFilter                        08c624ea077da91ce39ba0d36c951aa8
            Hlt1L0HadronFilterSequence                    5c7e5c693354d0abe25374cee28eb638
               Hlt1L0HadronDecision                       e81bf468f96274a7966fb3882d626c59
            Hlt1L0HadronPostScaler                        9b7f13ee778b7319e97f6719d87e65de
         Hlt1L0LocalPi0                                   0c21bf05dc2068de7ebdc48a45b05bbd
            Hlt1L0LocalPi0PreScaler                       da41c2814631ad1b00e97eee3b5f9ab8
            Hlt1L0LocalPi0L0DUFilter                      ef2241ca9f6e9ddbacca18688b6f8a67
            Hlt1L0LocalPi0FilterSequence                  03579590fe1eaeff532d8a7973de1161
               Hlt1L0LocalPi0Decision                     e9aa58d5c1edaabf3f2a716b1770723b
            Hlt1L0LocalPi0PostScaler                      94b24b282be7145ee203745a88f6f7f5
         Hlt1L0GlobalPi0                                  1e66b48e708ff540b2e796edfdf20faf
            Hlt1L0GlobalPi0PreScaler                      8d50f72ceca8fb0cab17f90509ed0db1
            Hlt1L0GlobalPi0L0DUFilter                     629cfb9c861aafc77d3d8e3b51aa4bd8
            Hlt1L0GlobalPi0FilterSequence                 a0d66cb6199202e119caf2b3ca4af6fa
               Hlt1L0GlobalPi0Decision                    606a0ca07d068475a0b979ea1f69a8fc
            Hlt1L0GlobalPi0PostScaler                     8b874842556353322d36d54d37078618
         Hlt1SingleHadron                                 651b7d50d13f98b07d8656e93d70060a
            Hlt1SingleHadronPreScaler                     2fc46380ef97da2c5f277c6f97c113f6
            Hlt1SingleHadronL0DUFilter                    fbf5f31c7b23ff7f7d2264983220c11e
            Hlt1SingleHadronFilterSequence                860438d2f617c8f7c4d63d8a5ec4be0f
               Hlt1L0AllHadronCandidates                  8142c5dc982556b494df77fb23a460fc
               Hlt1SingleHadronTFL0Hadrons                08709b6fd178db3cd9b9ef6f92d38bb1
               Hlt1RecoRZVeloSequence                     f702800c9b2879e503e04356081059c6
                  HltRecoRZVeloSequence                   72dca356288241f492d4a91a7db8719b
                     HltRecoRZVeloTracksSequence          6c5012951043410720681d588ac6995e
                        HltRecoRZVelo                     5daff62a5ec51f64000d94d7f52da70b
                     HltRecoRZPVSequence                  01e6c9d6a08d8bdc365789f94df4f747
                        HltRecoPV2D                       b5fde8cf8585a9ffb336106506e42ebf
                  Hlt1PrepareRZVelo                       1a0f5a1354db6982cd96dcf608cbf6e9
                  Hlt1PreparePV2D                         d5fd86bf04ece1813341ad663247913f
               Hlt1SingleHadronTFRZVelo                   94f07d6df39528fb5aeb75dca8475e7d
               Hlt1SingleHadronTUVelo                     b98c73720186cfaeda6988ae09609970
               Hlt1SingleHadronTF1Velo                    c5eb7ea4851fbffe867fc2ff88de80b1
               Hlt1SingleHadronTF2Velo                    6b7ebdb8e8e2753f0e7ddfeb40fef10c
               Hlt1SingleHadronTMVeloCalo                 e5218b4e0f8da60c1d66f30a69135fa7
```

| | | |
|---|---|---|
| Hlt1ElectronTrackWithIP | | cead6f7e56211651e369e0 |
| Hlt1ElectronTrackWithIPPreScaler | | 3898e4b3f80a4454ebc40a |
| Hlt1ElectronTrackWithIPL0DUFilter | | a6febdf2dbc02585e1b4b7 |
| Hlt1ElectronTrackWithIPFilterSequence | (*) | 4f5b84324604e22078f87d |
| Hlt1ElectronTrackWithIPTFL0Electrons | | e7b69d8b0224f81f278a2a |
| Hlt1ElectronTrackWithIPTUTConf | | d87d6e5f2400c2386d630f |
| Hlt1ElectronTrackWithIPTFRZVelo | | 450e98fc0cdc038b556b7c |
| Hlt1ElectronTrackWithIPTUVelo | | 8902b3e5b7362854ebca1b |
| Hlt1ElectronTrackWithIPTMVeloT | | 7ca16327afeb425751ea82 |
| Hlt1ElectronTrackWithIPTFVeloT | | 36f3ba78f932775a3d0999 |
| Hlt1ElectronTrackWithIPTFCompanionVelo | | d29c468c7f1722b63f7bb9 |
| Hlt1ElectronTrackWithIPTUCompanionForward | | 1ea8fa57b0c8400449a866 |
| Hlt1ElectronTrackWithIPTFCompanionForward | | bfb16635c3adba07f34ac3 |
| Hlt1ElectronTrackWithIPVM2DiElectron | | b3561d4d235928e36e1d15 |
| Hlt1ElectronTrackWithIPVFVertexCut | | e7665ebbc2b09485992619 |
| Hlt1ElectronTrackWithIPPostScaler | | e763b94a6870601c4686d9 |
| Hlt1ElectronFromPi0TrackWithIP | | e2a1e12b027280f31ba8fe |
| Hlt1ElectronFromPi0TrackWithIPPreScaler | | e6e27bc2b255629c0ecbcd |
| Hlt1ElectronFromPi0TrackWithIPL0DUFilter | | 7900220d71d1f0ca1446c7 |
| Hlt1ElectronFromPi0TrackWithIPFilterSequence | (*) | bd95d1fc26b3260f3b19b2 |
| Hlt1ElectronFromPi0TrackWithIPTFL0Electrons | | 023db24666b98c81e544e2 |
| Hlt1ElectronFromPi0TrackWithIPTUTConf | | 8075051512c27085a76d69 |
| Hlt1ElectronFromPi0TrackWithIPTFRZVelo | | 5a5d95a11d76ace1570a7c |
| Hlt1ElectronFromPi0TrackWithIPTUVelo | | 1becbab9b3c67d627cecd8 |
| Hlt1ElectronFromPi0TrackWithIPTMVeloT | | d017638e1c33aa37412aa9 |
| Hlt1ElectronFromPi0TrackWithIPTFVeloT | | a8c47440e7e653cd6fbc58 |
| Hlt1ElectronFromPi0TrackWithIPTFCompanionVelo | | 5b4889788304ded8d3f93b |
| Hlt1ElectronFromPi0TrackWithIPTUCompanionForward | | fed45a23a942f6bdfa58a0 |
| Hlt1ElectronFromPi0TrackWithIPTFCompanionForward | | f1fb2268579859871abc6 |
| Hlt1ElectronFromPi0TrackWithIPVM2DiElectron | | 7e6ee64f269181e65b6f68 |
| Hlt1ElectronFromPi0TrackWithIPVFVertexCut | | 495f950f13abf367ca7721 |
| Hlt1ElectronFromPi0TrackWithIPPostScaler | | 1acf0394fdab6cb59d1ee1 |

(*) Where did Hlt1L0{LocalPi0,Electron}Decision,Hlt1RecoRZVeloSequence go?

see https://savannah.cern.ch/bugs/?46919

- listProperties(0x803b0000,'Hlt1Global')
  - returns all properties of those algorithms whose name matches the regex ‘Hlt1Global’
- listProperties(0x803b0000,'Hlt1','Code|FilterDesc')
  - returns the properties whose name match ‘Code|FilterDesc’ for the algorithms which match the regex Hlt1

```
>>> listProperties(0x803b0000,'Hlt1ElectronFromPi0TrackWithIP','Code|FilterDesc|AcceptFraction')

   Requested Properties for DeterministicPrescaler/Hlt1ElectronFromPi0TrackWithIPPreScaler
      'AcceptFraction':1

   Requested Properties for LoKi::L0Filter/Hlt1ElectronFromPi0TrackWithIPL0DUFilter
      'Code':L0_CHANNEL('LocalPi0')

   Requested Properties for HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFL0Electrons
      'FilterDescriptor':[ 'L0ET,>,2600.' ]

   Requested Properties for HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFRZVelo
      'FilterDescriptor':[ 'RZVeloTMatch_Hlt1ElectronFromPi0TrackWithIPTUTConf,||<,60.' ]

   Requested Properties for HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFVeloT
      'FilterDescriptor':[ 'IP_PV2D,||[],0.1,3.' ]

   Requested Properties for HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFCompanionVelo
      'FilterDescriptor':[ 'IP_PV2D,||[],0.1,3.' , 'DOCA_Hlt1ElectronFromPi0TrackWithIPTFVeloT,<,0.2' ]

   Requested Properties for HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFCompanionForward
      'FilterDescriptor':[ 'PT,>,1000.0' ]

   Requested Properties for HltVertexMaker2/Hlt1ElectronFromPi0TrackWithIPVM2DiElectron
      'FilterDescriptor':[ 'DOCA,<,0.2' ]

   Requested Properties for HltVertexFilter/Hlt1ElectronFromPi0TrackWithIPVFVertexCut
      'FilterDescriptor':[ 'VertexPointing_PV2D,<,0.5' , 'VertexDz_PV2D,>,0.' ]

   Requested Properties for DeterministicPrescaler/Hlt1ElectronFromPi0TrackWithIPPostScaler
      'AcceptFraction':1
```

```
>>> listProperties(0x803b0000,'Hlt1ElectronTrackWithIPFilterSequence','Members')

   Requested Properties for GaudiSequencer/Hlt1ElectronTrackWithIPFilterSequence
      'Members':[ 'HltL0CaloCandidates/Hlt1L0ElectronDecision' ,
'HltTrackFilter/Hlt1ElectronTrackWithIPTFL0Electrons' ,
'HltTrackUpgrade/Hlt1ElectronTrackWithIPTUTConf' ,
'GaudiSequencer/Hlt1RecoRZVeloSequence' ,
'HltTrackFilter/Hlt1ElectronTrackWithIPTFRZVelo' ,
'HltTrackUpgrade/Hlt1ElectronTrackWithIPTUVelo' ,
'HltTrackMatch/Hlt1ElectronTrackWithIPTMVeloT' ,
'HltTrackFilter/Hlt1ElectronTrackWithIPTFVeloT' ,
'HltTrackUpgrade/Hlt1RecoVelo' ,
'HltTrackFilter/Hlt1ElectronTrackWithIPTFCompanionVelo' ,
'HltTrackUpgrade/Hlt1ElectronTrackWithIPTUCompanionForward' ,
'HltTrackFilter/Hlt1ElectronTrackWithIPTFCompanionForward' ,
'HltVertexMaker2/Hlt1ElectronTrackWithIPVM2DiElectron' ,
'HltVertexFilter/Hlt1ElectronTrackWithIPVFVertexCut' ]
```

```
>>> listProperties(0x803b0000,'Hlt1ElectronFromPi0TrackWithIPFilterSequence','Members')

   Requested Properties for GaudiSequencer/Hlt1ElectronFromPi0TrackWithIPFilterSequence
      'Members':[ 'HltL0CaloCandidates/Hlt1L0LocalPi0Decision' ,
'HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFL0Electrons' ,
'HltTrackUpgrade/Hlt1ElectronFromPi0TrackWithIPTUTConf' ,
'GaudiSequencer/Hlt1RecoRZVeloSequence' ,
'HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFRZVelo' ,
'HltTrackUpgrade/Hlt1ElectronFromPi0TrackWithIPTUVelo' ,
'HltTrackMatch/Hlt1ElectronFromPi0TrackWithIPTMVeloT' ,
'HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFVeloT' ,
'HltTrackUpgrade/Hlt1RecoVelo' ,
'HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFCompanionVelo' ,
'HltTrackUpgrade/Hlt1ElectronFromPi0TrackWithIPTUCompanionForward' ,
'HltTrackFilter/Hlt1ElectronFromPi0TrackWithIPTFCompanionForward' ,
'HltVertexMaker2/Hlt1ElectronFromPi0TrackWithIPVM2DiElectron' ,
'HltVertexFilter/Hlt1ElectronFromPi0TrackWithIPVFVertexCut' ]
>>>
```

- getRoutingBits(0x803b0000) / listRoutingBits(0x803b0000)
  - returns / list a (python) list of Routing Bit definitions, given either a TCK or a configID

```
>>> listRoutingBits(0x801b0000)                    <-- Moore v5r5 configuration
{32: 'Hlt1Global', 33: 'Hlt1LumiDecision', 34: 'Hlt1PhysicsDecision|Hlt1RandomDecision|Hlt1VeloASideDecision|Hlt1VeloCSideDecision|
Hlt1SingleMuonNoIPDecision|Hlt1SingleMuonIPandPTDecision|Hlt1DiMuonFromL0DiMuoNoIPDecision|
Hlt1DiMuonFromL0DiMuonIPDecision|Hlt1DiMuonFrom2L0IPDecision|Hlt1DiMuonFrom2L0NoIPDecision|
Hlt1DiMuonFromMuonSegIPDecision|Hlt1DiMuonFromMuonSegNoIPDecision|Hlt1MuonHadronDecision|Hlt1SingleHadronDecision|
Hlt1DiHadronDecision|Hlt1PhotonDecision|Hlt1SingleElectronDecision|Hlt1ElectronTrackDecision', 35: 'Hlt1VeloCSideDecision|
Hlt1VeloASideDecision', 36: 'Hlt1RandomDecision', 37: 'Hlt1PhysicsDecision'}
>>>
>>>
>>> listRoutingBits(0x803b0000)                    <-- Moore v6r1 configuration
{32: "HLT_PASS('Hlt1Global')", 33: "HLT_PASS('Hlt1LumiDecision')", 34: "HLT_PASS('Hlt1IgnoringLumiDecision')", 35:
"HLT_PASS('Hlt1VeloClosingDecision')", 36: "HLT_PASS('Hlt1XPressDecision')"}
>>>
```

TODO : make HLT_PASS and HLT_PASSIGNORING accept wildcards, so we can write the above as:

{32: "HLT_PASS('Hlt1Global')", 33: "HLT_PASS('Hlt1Lumi.*Decision')", 34: "HLT_PASSIGNORING('Hlt1Lumi.*Decision')", 35:
"HLT_PASS('Hlt1Velo.*Decision')", 36: "HLT_PASS('Hlt1XPressDecision')"}

which will get rid of the need to create dedicated lines which are the 'or' of others (and thus introduce ordering constraints!) just to keep this legible...

- diff(0x80350000,0x803b0000)
  - shows items only in left, only in right, and differences of items in both
  - *(still!) TODO: should* be used internally before allowing the addition of a configuration to an existing hltType -- if 'only in' appears, should be forbidden (as it implies a change in the 'flow')
- updateProperties( ... )
  - allows to copy-and-modify existing configurations (eg. change prescales)
- createTCKentries( { .. : .. , ..... } )
  - creates entries in mapping from TCK to configID
  - eg. update to new L0 TCK, but keep same HLT config

# The configuration mistakes fixed during the 2nd FEST week...

```
>>> diff(0x80350000,0x803b0000)
--- DeterministicPrescaler/Hlt1VeloASidePreScaler (IAlgorithm) d6d84499f22b9d49b170690ed272be40
+++ DeterministicPrescaler/Hlt1VeloASidePreScaler (IAlgorithm) 7843022e9c4f5b88678b790f7318b3a0
@@ -28,1 +28,1 @@
-'AcceptFraction':0.0001
+'AcceptFraction':1

--- DeterministicPrescaler/Hlt1VeloCSidePreScaler (IAlgorithm) d6d84499f22b9d49b170690ed272be40
+++ DeterministicPrescaler/Hlt1VeloCSidePreScaler (IAlgorithm) 7843022e9c4f5b88678b790f7318b3a0
@@ -28,1 +28,1 @@
-'AcceptFraction':0.0001
+'AcceptFraction':1

--- DeterministicPrescaler/LumiStripperPrescaler (IAlgorithm) d6d84499f22b9d49b170690ed272be40
+++ DeterministicPrescaler/LumiStripperPrescaler (IAlgorithm) 7843022e9c4f5b88678b790f7318b3a0
@@ -28,1 +28,1 @@
-'AcceptFraction':0
+'AcceptFraction':0.9999

--- HltRoutingBitsWriter/HltRoutingBitsWriter (IAlgorithm) d6d84499f22b9d49b170690ed272be40
+++ HltRoutingBitsWriter/HltRoutingBitsWriter (IAlgorithm) 7843022e9c4f5b88678b790f7318b3a0
@@ -16,1 +16,1 @@
-'RoutingBits':{ 32 : "HLT_PASS('Hlt1Global')" , 33 : "HLT_PASS('Hlt1LumiDecision')" , 34 :
"HLT_PASS('Hlt1IgnoringLumiDecision')" , 35 : "HLT_PASS('Hlt1VeloClosingDecision')" , 36 :
"HLT_PASS('Hlt1ExpressDecision')" }
+'RoutingBits':{ 32 : "HLT_PASS('Hlt1Global')" , 33 : "HLT_PASS('Hlt1LumiDecision')" , 34 :
"HLT_PASS('Hlt1IgnoringLumiDecision')" , 35 : "HLT_PASS('Hlt1VeloClosingDecision')" , 36 :
"HLT_PASS('Hlt1XPressDecision')" }

>>>
```

First three now solved by overruling HltConf defaults in dedicated ‘FEST’ python file in Moore

# Other items (still) on the list...

- move online Moore setup over to start Moore as python, not .opts....
  - more and more options files are disappearing, and being replaced by .py
  - eg. $STDOPTS/DecodeRawEvent.opts
  - Need to remove any dependencies on 'external' .opts
  - have an example from lumi to follow
- verify independence of HltLines
  - run 'everything together', record result for each line, then run 'one line at a time' job, compare output, fix discrepancies
    - can now run 'one at a time' easily, but need to automate *which* one to run (and loop over them)..
- Move Hlt2 to use HltLine, and additional pythonization of Hlt2
- Allow slightly selective disable histogramming

# Other items on the list...

- Switch to use DB snapshots (ready in principle, but needs testing)
- Conditions updates at fast-run change

- verify TCK changes at fast-run:
  - not all algorithms support updating of their properties after initialize
    - missing appropriate updateHandler, or (complex cases only) explicit 'restart'
  - verify by using the 'wrong' TCK during initialize, causes restart on first event, compare with using the 'right' TCK during init
  - need to 'sign off' / whitelist those that are known to work
    - DeterministicPrescaler probably the one we need first, it is known to work properly ;-)
    - LoKi based filters also OK (used as ODIN filter, L0DU filter and HltDecReports filter in Hlt1)
    - The above probably cover virtually everything really needed in practice...